CHAPTER 15

Cost Sharing

Kamal Jain and Mohammad Mahdian

Abstract

The objective of cooperative game theory is to study ways to enforce and sustain cooperation among agents willing to cooperate. A central question in this field is how the benefits (or costs) of a joint effort can be divided among participants, taking into account individual and group incentives, as well as various fairness properties.

In this chapter, we define basic concepts and review some of the classical results in the cooperative game theory literature. Our focus is on games that are based on combinatorial optimization problems such as facility location. We define the notion of cost sharing, and explore various incentive and fairness properties cost-sharing methods are often expected to satisfy. We show how cost-sharing methods satisfying a certain property termed cross-monotonicity can be used to design mechanisms that are robust against collusion, and study the algorithmic question of designing cross-monotonic cost-sharing schemes for combinatorial optimization games. Interestingly, this problem is closely related to linear-programming-based techniques developed in the field of approximation algorithms. We explore this connection, and explain a general method for designing cross-monotonic cost-sharing schemes, as well as a technique for proving impossibility bounds on such schemes. We will also discuss an *axiomatic* approach to characterize two widely applicable solution concepts: the Shapley value for cooperative games, and the Nash bargaining solution for a more restricted framework for surplus sharing.

15.1 Cooperative Games and Cost Sharing

Consider a setting where a set $\mathcal{A}$ of n agents seek to cooperate in order to generate value. The value generated depends on the coalition S of agents cooperating. In general, the set of possible outcomes of cooperation among agents in $S \subseteq \mathcal{A}$ is denoted by $V(S)$, where each outcome is given by a vector in $\mathbb{R}^S$, whose i'th component specifies the utility that the agent $i \in S$ derives in this outcome. The set $\mathcal{A}$ of agents along with the function V defines what is called a *cooperative game* (also known as a *coalitional game*) *with nontransferable utilities* (abbreviated as an *NTU game*). A special case, called a *cooperative game with transferable utilities* (abbreviated as a *TU game*), is when the

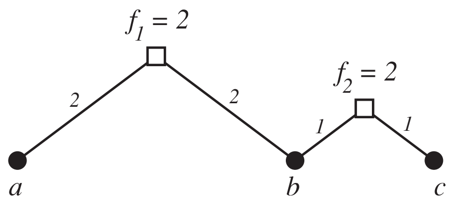

Figure 15.1. An example of the facility location game.

value generated by a coalition can be divided in an arbitrary way among the agents in S. In other words, a TU game is defined by specifying a function v: $2^{\mathcal{A}} \mapsto \mathbb{R}$, which gives the value $v(S) \in \mathbb{R}$ generated by each coalition S. We assume $v(\emptyset) = 0$. The set of all possible outcomes in such a game is defined as $V(S) = \{x \in \mathbb{R}^S \colon \sum_{i \in S} x_i \le v(S)\}$.

The notion of a cooperative game was first proposed by von Neumann and Morgenstern. This notion seeks to abstract away all other aspects of the game except the combinatorial aspect of the coalitions that can form. This is in contrast with noncooperative games, where the focus is on the set of choices (moves) available to each agent.

Note that in the definition of a cooperative game, we did not restrict the values to be nonnegative.[1] In fact, the case that all values are nonpositive is the focus of this chapter, as it corresponds to the problem of sharing the cost of a service among those who receive the service (this is by taking the value to be the negative of the cost). Again, the cost-sharing problem can be studied in both the TU and the NTU models. The TU model applies to settings where, for example, a service provider incurs some (monetary) cost $c(S)$ in building a network that connects a set S of customers to the Internet, and needs to divide this cost among customers in S. In practice, the cost function c is often defined by solving a combinatorial optimization problem. One example, which we will use throughout the chapter, is the facility location game defined below.

Definition 15.1 In the *facility location game*, we are given a set $\mathcal{A}$ of agents (also known as cities, clients, or demand points), a set $\mathcal{F}$ of facilities, a facility opening cost f_i for every facility $i \in \mathcal{F}$, and a distance d_{ij} between every pair (i, j) of points in $\mathcal{A} \cup \mathcal{F}$ indicating the cost of connecting j to i. We assume that the distances come from a metric space; i.e., they are symmetric and obey the triangle inequality. For a set $S \subseteq \mathcal{A}$ of agents, the cost of this set is defined as the minimum cost of opening a set of facilities and connecting every agent in S to an open facility. More precisely, the cost function c is defined by $c(S) = \min_{\mathcal{F}' \subseteq \mathcal{F}} \{\sum_{i \in \mathcal{F}'} f_i + \sum_{j \in S} \min_{i \in \mathcal{F}'} d_{ij}\}$.

Example 15.2 Figure 15.1 shows an instance of the facility location game with 3 agents $\{a, b, c\}$ and 2 facilities $\{1, 2\}$. The distances between some pairs are marked in the figure, and other distances can be calculated using the triangle

[1] If all values are nonnegative, the problem is called a *surplus sharing* problem.

inequality (e.g., the distance between facility 1 and client c is $2 + 1 + 1 = 4$). The cost function defined by this instance is the following:

$$c(\{a\}) = 4, \quad c(\{b\}) = 3, \quad c(\{c\}) = 3,$$
$$c(\{a, b\}) = 6, \quad c(\{b, c\}) = 4, \quad c(\{a, c\}) = 7, \quad c(\{a, b, c\}) = 8.$$

Since the monetary cost may be distributed arbitrarily among the agents, it is natural to model the above example as a TU game. An example of a case where the NTU model is more applicable is a network design problem where the cost incurred by an agent i in the set of agents S receiving the service corresponds to the delay this agent suffers. There are multiple designs for the network connecting the customers in S, and each design corresponds to a profile of delays that these agents will suffer. The set of possible outcomes for the coalition S is defined as the collection of all such profiles, and is denoted by $C(S)$. As delays are nontransferrable, this setting is best modeled as an NTU cost-sharing game. For another example of an NTU game, see the housing allocation problem in Section 10.3 of this book.

As most of the work on cost sharing in the algorithmic game theory literature has so far focused on TU games, this chapter is mainly devoted to such games; henceforth, by a cost-sharing game we mean a TU game, unless otherwise stated.

15.2 Core of Cost-Sharing Games

A central notion in cooperative game theory is the notion of *core*. Roughly speaking, the core of a cooperative game is an outcome of cooperation among all agents where no coalition of agents can all benefit by breaking away from the grand coalition. Intuitively, the core of a game corresponds to situations where it is possible to sustain cooperation among all agents in an economically stable manner.

In this section, we define the notion of core for cost-sharing games, and present two classical results on conditions for nonemptiness of the core. We show how the notion of core for TU games can be relaxed to an approximate version suitable for hard combinatorial optimization games, and observe a connection between this notion and the integrality gap of a linear programming relaxation of such problems.

15.2.1 Core of TU Games

Formally, the core of a TU cost-sharing game is defined as follows.

Definition 15.3 Let $(\mathcal{A}, c)$ be a TU cost-sharing game. A vector $\boldsymbol{\alpha} \in \mathbb{R}^{\mathcal{A}}$ (sometimes called a *cost allocation*) is in the *core* of this game if it satisfies the following two conditions:

- **Budget balance:** $\sum_{j \in \mathcal{A}} \alpha_j = c(\mathcal{A})$.
- **Core property:** for every $S \subseteq \mathcal{A}$, $\sum_{j \in S} \alpha_j \leq c(S)$.

Example 15.4 As an example, consider the facility location game of Example 15.2 (Figure 15.1). It is not hard to verify that the vector $(4, 2, 2)$ lies in the core of this game. In fact, this is not the only cost allocation in the core of this game; for example, (4, 1, 3) is also in the core. On the other hand, if a third facility with opening cost 3 and distance 1 to agents a and c is added to this game, the resulting game will have an empty core. To see this, note that after adding the third facility, we have $c(\{a, c\}) = 5$. Now, if there is a vector $\boldsymbol{\alpha}$ in the core of this game, we must have

$$\begin{aligned} \alpha_a + \alpha_b &\leq c(\{a, b\}) = 6 \\ \alpha_b + \alpha_c &\leq c(\{b, c\}) = 4 \\ \alpha_a + \alpha_c &\leq c(\{a, c\}) = 5 \end{aligned}$$

By adding the above three inequalities and dividing both sides by 2, we obtain $\alpha_a + \alpha_b + \alpha_c \leq 7.5 < c(\{a, b, c\})$. Therefore $\boldsymbol{\alpha}$ cannot be budget balanced.

A classical result in cooperative game theory, known as the Bondareva–Shapley theorem, gives a necessary and sufficient condition for a game to have nonempty core. To state this theorem, we need the following definition.

Definition 15.5 A vector $\boldsymbol{\lambda}$ that assigns a nonnegative weight λ_S to each subset $S \subseteq \mathcal{A}$ is called a *balanced collection of weights* if for every $j \in \mathcal{A}$, $\sum_{S:j\in S} \lambda_S = 1$.

Theorem 15.6 *A cost-sharing game $(\mathcal{A}, c)$ with transferable utilities has a nonempty core if and only if for every balanced collection of weights $\boldsymbol{\lambda}$, we have $\sum_{S\subseteq\mathcal{A}} \lambda_S c(S) \geq c(\mathcal{A})$.*

PROOF By the definition of the core, the game $(\mathcal{A}, c)$ has a nonempty core if and only if the solution of the following linear program (LP) is precisely $c(\mathcal{A})$. (Note that this solution can never be larger than $c(\mathcal{A})$.)

$$\begin{aligned} &\text{Maximize} && \textstyle\sum_{j\in\mathcal{A}} \alpha_j \\ &\text{Subject to} && \forall S \subseteq \mathcal{A} : \textstyle\sum_{j\in S} \alpha_j \leq c(S). \end{aligned} \tag{15.1}$$

By strong LP duality, the solution of the above LP is equal to the solution of the following dual program:

$$\begin{aligned} &\text{Minimize} && \sum_{S\subseteq\mathcal{A}} \lambda_S c(S) \\ &\text{Subject to} && \forall j \in \mathcal{A} : \sum_{S:j\in S} \lambda_S = 1 \\ &&& \forall S \subseteq \mathcal{A} : \lambda_S \geq 0. \end{aligned} \tag{15.2}$$

Therefore, the core of the game is nonempty if and only if the solution of the LP (15.2) is equal to $c(\mathcal{A})$. By definition, feasible solutions of this program are balanced collections of weights. Therefore, the core of the game $(\mathcal{A}, c)$ is nonempty if and only if for every balanced collection of weights (λ_S), $\sum_{S\subseteq\mathcal{A}} \lambda_S c(S) \geq c(\mathcal{A})$. □

As an example, the proof of emptiness of the core given in Example 15.4 can be restated by defining a vector $\boldsymbol{\lambda}$ as follows: $\lambda_{\{a,b\}} = \lambda_{\{b,c\}} = \lambda_{\{a,c\}} = \frac{1}{2}$ and $\lambda_S = 0$ for every other set S. It is easy to verify that $\boldsymbol{\lambda}$ is a balanced collection of weights and $\sum_{S \subseteq \mathcal{A}} \lambda_S c(S) < c(\mathcal{A})$.

15.2.2 Approximate Core

As we saw in Example 15.4, a difficulty with the notion of core is that the core of many cost-sharing games, including most combinatorial optimization games based on computationally hard problems, is often empty. Furthermore, when the underlying cost function is hard to compute (e.g., in the facility location game), even deciding whether the core of the game is empty is often computationally intractable. This motivates the following definition.

Definition 15.7 A vector $\boldsymbol{\alpha} \in \mathbb{R}^{\mathcal{A}}$ is in the γ-approximate core (or γ-core, for short) of the game $(\mathcal{A}, c)$ if it satisfies the following two conditions:

- **γ-Budget balance:** $\gamma c(\mathcal{A}) \leq \sum_{j \in \mathcal{A}} \alpha_j \leq c(\mathcal{A})$.
- **Core property:** for every $S \subseteq \mathcal{A}$, $\sum_{j \in S} \alpha_j \leq c(S)$.

For example, in the facility location game given in Example 15.4, the vector $(3.5, 2.5, 1.5)$ is in the $\frac{7.5}{8}$-core of the game. Note that the argument given to show the emptiness of the core of this game actually proves that for every $\gamma > \frac{7.5}{8}$, the γ-core of this game is empty.

The Bondareva–Shapley theorem can be easily generalized to the following approximate version.

Theorem 15.8 *For every $\gamma \leq 1$, a cost-sharing game $(\mathcal{A}, c)$ with transferable utilities has a nonempty γ-core if and only if for every balanced collection of weights $\boldsymbol{\lambda}$, we have $\sum_{S \subseteq \mathcal{A}} \lambda_S c(S) \geq \gamma c(\mathcal{A})$.*

The proof is similar to the proof of the Bondareva–Shapley theorem and is based on the observation that by LP duality, the γ-core of the game is nonempty if and only if the solution of the LP (15.2) is at least $\gamma c(S)$. Note that if the cost function c is subadditive (i.e., $c(S_1 \cup S_2) \leq c(S_1) + c(S_2)$ for any two disjoint sets S_1 and S_2), then the optimal *integral* solution of the LP (15.2) is precisely $c(\mathcal{A})$. Therefore,

Corollary 15.9 *For any cost-sharing game $(\mathcal{A}, c)$ with a subadditive cost function, the largest value γ for which the γ-core of this game is nonempty is equal to the integrality gap of the LP (15.2).*

As it turns out, for many combinatorial optimization games such as set cover, vertex cover, and facility location, the LP formulation (15.2) is in fact equivalent to the standard (polynomial-size) LP formulation of the problem, and hence Corollary 15.9 translates into a statement about the integrality gap of the standard LP formulation of the problem. Here, we show this connection for the facility location game.

We start by formulating the facility location problem as an integer program. In this formulation, x_i and y_{ij} are variables indicating whether facility i is open, and whether agent j is connected to facility i.

$$\begin{array}{ll} \text{Minimize} & \sum_{i \in \mathcal{F}} f_i x_i + \sum_{i \in \mathcal{F}} \sum_{j \in \mathcal{A}} d_{ij} y_{ij} \\ \text{Subject to} & \forall j \in \mathcal{A} : \sum_{i \in \mathcal{F}} y_{ij} \geq 1 \\ & \forall i \in \mathcal{F}, j \in \mathcal{A} : x_i \geq y_{ij} \\ & \forall i \in \mathcal{F}, j \in \mathcal{A} : x_i, y_{ij} \in \{0, 1\}. \end{array} \tag{15.3}$$

By relaxing the second constraint to $x_i, y_{ij} \geq 0$ we obtain an LP whose dual can be written as follows:

$$\begin{array}{ll} \text{Maximize} & \sum_{j \in \mathcal{A}} \alpha_j \\ \text{Subject to} & \forall i \in \mathcal{F}, j \in \mathcal{A} : \beta_{ij} \geq \alpha_j - d_{ij} \\ & \forall i \in \mathcal{F} : \sum_{j \in \mathcal{A}} \beta_{ij} \leq f_i \\ & \forall i \in \mathcal{F}, j \in \mathcal{A} : \alpha_j, \beta_{ij} \geq 0 \end{array} \tag{15.4}$$

Note that we may assume without loss of generality that in a feasible solution of the above LP, $\beta_{ij} = \max(0, \alpha_j - d_{ij})$. Thus, to specify a dual solution it is enough to give $\boldsymbol{\alpha}$. We now observe that the above dual LP captures the core constraint of the facility location game; i.e., it is equivalent to the LP (15.1).

Proposition 15.10 *For any feasible solution* $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ *of the LP (15.4),* $\boldsymbol{\alpha}$ *satisfies the core property of the facility location game.*

PROOF We need to show that for every set $S \subseteq \mathcal{A}$, $\sum_{j \in S} \alpha_j \leq c(S)$, where $c(S)$ is the cost of the facility location problem for agents in S. First we note that for any facility i and set of agents $R \subseteq \mathcal{A}$, by adding the first and the second inequalities of the LP (15.4) for facility i and every $j \in R$ we obtain

$$\sum_{j \in R} \alpha_j \leq f_i + \sum_{j \in R} d_{ij}. \tag{15.5}$$

Now, consider an optimal solution for the set of agents S, and assume $i_1, \ldots, i_k$ are facilities that are open and R_ℓ is the set of agents served by facility i_ℓ in this solution. Summing Inequality (15.5) for every (i_ℓ, R_ℓ) will yield the result. □

By the above proposition, the solution of the dual LP (15.4) (which, by LP duality, is the same as the LP relaxation of (15.3)) is equal to the solution of the LPs (15.1) and (15.2). Furthermore, the optimal integral solution of (15.3) is $c(\mathcal{A})$. Therefore, the integrality gap of (15.3) is the same as that of (15.2) and gives the best budget balance factor that a cost allocation satisfying the core property can achieve. The best known results in the field of approximation algorithms show that this gap (in the worst case) is between $\frac{1}{1.52}$ and $\frac{1}{1.463}$.

15.2.3 Core of NTU Games

We conclude this section with a classical theorem due to Scarf, which gives a sufficient condition for the nonemptiness of the core of NTU games similar to the one given in Theorem 15.6 for TU games. However, unlike in the case of TU games, the condition given in Scarf's theorem is not necessary for the nonemptiness of the core.

Formally, the core of an NTU cost-sharing game $(\mathcal{A}, C)$ is the collection of all cost allocations $\boldsymbol{\alpha} \in C(\mathcal{A})$ such that there is no nonempty coalition $S \subseteq \mathcal{A}$ and cost allocation $\mathbf{x} \in C(S)$ for which $x_j < \alpha_j$ for all $j \in S$. Note that this definition coincides with Definition 15.3 in the case of a TU game. In the following theorem the *support* of a balanced collection of weights $\boldsymbol{\lambda}$ denotes the collection of all sets S with $\lambda_S > 0$.

Theorem 15.11 *Let $(\mathcal{A}, C)$ be a cost-sharing game with nontransferable utilities. Assume for every balanced collection of weights $\boldsymbol{\lambda}$ and every vector $\mathbf{x} \in \mathbb{R}^{\mathcal{A}}$ the following property holds: if for every set S in the support of $\boldsymbol{\lambda}$, the restriction of $\mathbf{x}$ to the coordinates in S is in $C(S)$, then $\mathbf{x} \in C(\mathcal{A})$. Then $(\mathcal{A}, C)$ has a nonempty core.*

The proof of the above theorem, which is beyond the scope of this chapter, uses an adaptation of the Lemke–Howson algorithm for computing Nash equilibria (described in Section 3.4 of this book), and is considered an early and important contribution of the algorithmic viewpoint in game theory. However, the worst case running time of this algorithm (like the Lemke–Howson algorithm) is exponential in $|\mathcal{A}|$. This is in contrast to the proof of the Bondareva–Shapley theorem, which gives a polynomial-time algorithm[2] for computing a point in the core of the game, if the core is nonempty.

15.3 Group-Strategyproof Mechanisms and Cross-Monotonic Cost-Sharing Schemes

The cost-sharing problem defined in this chapter models the pricing problem for a service provider with a given set of customers. In settings where the demand is sensitive to the price, an alternative choice for the service provider is to conduct an auction between the potential customers to select the set of customers who can receive the service based on their willingness to pay and the cost structure of the problem. The goal is to design an auction mechanism that provides incentives for individuals as well as groups of agents to bid truthfully. In this section, we study this problem and exhibit its connection to cost sharing.

We start with the definition of the setting. Let $\mathcal{A}$ be a set of n agents interested in receiving a service. The cost of providing service is given by a cost function c: $2^{\mathcal{A}} \mapsto \mathbb{R}^+ \cup \{0\}$, where $c(S)$ specifies the cost of providing service for agents in S. Each agent i has a value $u_i \in \mathbb{R}$ for receiving the service; that is, she is willing to pay at most u_i to get the service. We further assume that the utility of agent i is given by $u_i q_i - x_i$,

[2] This is assuming a suitable representation for the cost function c, e.g., by a separation oracle for (15.1), or in combinatorial optimization games where statements like Proposition 15.10 hold.

where q_i is an indicator variable that indicates whether she has received the service or not, and x_i is the amount she has to pay. A *cost-sharing mechanism* is an algorithm that elicits a bid $b_i \in \mathbb{R}$ from each agent, and based on these bids, decides which agents should receive the service and how much each of them has to pay. More formally, a cost-sharing mechanism is a function that associates to each vector $\mathbf{b}$ of bids a set $Q(\mathbf{b}) \subseteq \mathcal{A}$ of agents to be serviced, and a vector $\mathbf{p}(\mathbf{b}) \in \mathbb{R}^n$ of payments. When there is no ambiguity, we write Q and $\mathbf{p}$ instead of $Q(\mathbf{b})$ and $\mathbf{p}(\mathbf{b})$, respectively. We assume that a mechanism satisfies the following conditions:

- *No Positive Transfer (NPT)*: The payments are nonnegative (i.e., $p_i \geq 0$ for all i).
- *Voluntary Participation (VP)*: An agent who does not receive the service is not charged (i.e., $p_i = 0$ for $i \notin Q$), and an agent who receives the service is not charged more than her bid (i.e., $p_i \leq b_i$ for $i \in Q$)
- *Consumer Sovereignty (CS)*: For each agent i, there is some bid b_i^* such that if i bids b_i^*, she will get the service, no matter what others bid.

Furthermore, we want the mechanisms to be approximately budget-balanced. We call a mechanism γ-budget-balanced with respect to the cost function c if the total amount the mechanism charges the agents is between $\gamma c(Q)$ and $c(Q)$ (i.e., $\gamma c(Q) \leq \sum_{i \in Q} x_i \leq c(Q)$).

We look for mechanisms, called *group strategyproof mechanisms*, which satisfy the following property in addition to NPT, VP, and CS. Let $S \subseteq \mathcal{A}$ be a coalition of agents, and $\mathbf{u}, \mathbf{u}'$ be two vectors of bids satisfying $u_i = u_i'$ for every $i \notin S$ (we think of $\mathbf{u}$ as the values of agents, and $\mathbf{u}'$ as a vector of strategically chosen bids). Let $(Q, \mathbf{p})$ and $(Q', \mathbf{p}')$ denote the outputs of the mechanism when the bids are $\mathbf{u}$ and $\mathbf{u}'$, respectively. A mechanism is *group strategyproof* if for every coalition S of agents, if the inequality $u_i q_i' - p_i' \geq u_i q_i - p_i$ holds for every $i \in S$, then it holds with equality for every $i \in S$. In other words, there should not be any coalition S and vector $\mathbf{u}'$ of bids such that if members of S announce $\mathbf{u}'$ instead of $\mathbf{u}$ (their true value) as their bids, then every member of the coalition S is at least as happy as in the truthful scenario, and at least one person is happier.

Moulin showed that cost-sharing methods satisfying an additional property termed *cross-monotonicity* can be used to design group-strategyproof cost-sharing mechanisms. The cross-monotonicity property captures the notion that agents should not be penalized as the serviced set grows. To define this property, we first need to define the notion of a cost-sharing *scheme*.

Definition 15.12 Let $(\mathcal{A}, c)$ denote a cost-sharing game. A *cost-sharing scheme* is a function that for each set $S \subseteq \mathcal{A}$, assigns a cost allocation for S. More formally, a cost-sharing scheme is a function ξ: $\mathcal{A} \times 2^{\mathcal{A}} \mapsto \mathbb{R}$ such that, for every $S \subseteq \mathcal{A}$ and every $i \notin S$, $\xi(i, S) = 0$. We say that a cost-sharing scheme ξ is γ-budget balanced if for every set $S \subseteq \mathcal{A}$, we have $\gamma c(S) \leq \sum_{i \in S} \xi(i, S) \leq c(S)$.

Definition 15.13 A cost-sharing scheme ξ is *cross-monotone* if for all $S, T \subseteq \mathcal{A}$ and $i \in S$, $\xi(i, S) \geq \xi(i, S \cup T)$.

```
Mechanism $\mathcal{M}_\xi$
    Initialize $S \leftarrow \mathcal{A}$.
    Repeat
        Let $S \leftarrow \{i \in S : b_i \geq \xi(i, S)\}$.
    Until for all $i \in S$, $b_i \geq \xi(i, S)$.
    Return $Q = S$ and $p_i = \xi(i, S)$ for all $i$.
```

Figure 15.2. Moulin's group-strategyproof mechanism.

The following proposition shows that cross-monotonicity is a stronger property than core.

Proposition 15.14 *Let ξ be an γ-budget-balanced cross-monotonic cost sharing scheme for the cost-sharing game $(\mathcal{A}, c)$. Then $\xi(., \mathcal{A})$ is in the γ-core of this game.*

PROOF We need to verify only that $\xi(., \mathcal{A})$ satisfies the core property, i.e., for every set $S \subseteq \mathcal{A}$, $\sum_{i \in S} \xi(i, \mathcal{A}) \leq c(S)$. By the cross-monotonicity property, for every $i \in S$, $\xi(i, \mathcal{A}) \leq \xi(i, S)$. Therefore, $\sum_{i \in S} \xi(i, \mathcal{A}) \leq \sum_{i \in S} \xi(i, S) \leq c(S)$, where the last inequality follows from the γ-budget balance property of ξ. □

Given a cross-monotonic cost-sharing scheme ξ for the cost-sharing game $(\mathcal{A}, c)$, we define a cost-sharing mechanism $\mathcal{M}_\xi$ as presented in Figure 15.2.

The following proposition provides an alternative way to view the mechanism $\mathcal{M}_\xi$.

Proposition 15.15 *Assume ξ is a cross-monotonic cost sharing scheme for the cost-sharing game $(\mathcal{A}, c)$, and $b_i \in \mathbb{R}^+ \cup \{0\}$ for every $i \in \mathcal{A}$. Then there is a unique maximal set $S \subseteq \mathcal{A}$ satisfying the property that for every $i \in S$, $b_i \geq \xi(i, S)$. The mechanism $\mathcal{M}_\xi$ returns this set.*

PROOF Assume that two different maximal sets S_1 and S_2 satisfy the stated property, i.e., $b_i \geq \xi(i, S_1)$ for every $i \in S_1$ and $b_i \geq \xi(i, S_2)$ for every $i \in S_2$. Then for every $i \in S_1$, $b_i \geq \xi(i, S_1) \geq \xi(i, S_1 \cup S_2)$, where the last inequality follows from cross-monotonicity of ξ. Similarly, for every $i \in S_2$, $b_i \geq \xi(i, S_1 \cup S_2)$. Therefore, the set $S_1 \cup S_2$ also satisfies this property. This contradicts with the maximality of S_1 and S_2.

Let S^* denote the unique maximal set satisfying $b_i \geq \xi(i, S^*)$ for all $i \in S^*$. We claim that $\mathcal{M}_\xi$ never eliminates any of the agents in S^* from the serviced set S. Consider, for contradiction, the first step where it eliminates an agent $i \in S^*$ from the serviced set S. This means that we must have $b_i < \xi(i, S)$. However, since $S^* \subseteq S$, by cross-monotonicity we have $\xi(i, S) \leq \xi(i, S^*)$, and hence $b_i < \xi(i, S^*)$, contradicting the definition of S^*. Therefore, the set Q returned by $\mathcal{M}_\xi$ contains S^*. By maximality of S^*, it cannot contain any other agent, that is, $Q = S^*$. □

We are now ready to prove the following theorem of Moulin.

Theorem 15.16 *If ξ is an γ-budget-balanced cross-monotonic cost-sharing scheme, then $\mathcal{M}_\xi$ is group-strategyproof and γ-budget balanced.*

PROOF Assume, for contradiction, that there is a coalition T of agents that benefits from bidding according to the vector $\mathbf{u}'$ instead of their true values $\mathbf{u}$. Agents in T can be partitioned into two sets T^+ and T^- based on whether their bid in $\mathbf{u}'$ is greater than their bid in $\mathbf{u}$, or not. First, we claim that it can be assumed, without loss of generality, that T^+ is empty, i.e., agents cannot benefit from overbidding. To see this, we start from a bid vector where every agent in T bids according to $\mathbf{u}'$ (and others bid truthfully), and reduce the bids of the agents in T^+ to their true value one by one. If at any step, e.g., when the bid of agent $i \in T^+$ is reduced from u'_i to u_i, the outcome of the auction changes, then by Proposition 15.15, i must be a winner when she bids according to $\mathbf{u}'$, and not a winner when she bids according to $\mathbf{u}$. This means that $u'_i \geq \xi(i, S_i) > u_i$, where S_i is the set of winners when i bids according to $\mathbf{u}'$. However, this means that the agent i must pay an amount greater than her true value in the scenario where every agent in T bids according to $\mathbf{u}'$. This is in contradiction with the assumption that agents in T all benefit from the collusion. By this argument, the bid of every agent in T^+ can be lowered to her true value without changing the outcome of the auction. Therefore, we assume without loss of generality that T^+ is empty.

Now, let S' and S denote the set of winners in the untruthful and the truthful scenarios (i.e., when agents bid according to $\mathbf{u}'$ and $\mathbf{u}$), respectively. As the bid of each agent in $\mathbf{u}'$ is less than or equal to her bid in $\mathbf{u}$, by Proposition 15.15, $S' \subseteq S$. By cross-monotonicity, this implies that the payment of each agent in the untruthful scenario is at least as much as her payment in the truthful scenario. Therefore, no agent can be strictly happier in the untruthful scenario than in the truthful scenario. □

Moulin's theorem shows that cross-monotonic cost-sharing schemes give rise to group-strategyproof mechanisms. An interesting question is whether the converse also holds, i.e., is there a way to construct a cross-monotonic cost-sharing scheme given a group-strategyproof mechanism? The answer to this question is negative (unless the cost function is assumed to be submodular), as there are examples where a cost function has a group-strategyproof mechanism but no cross-monotonic cost-sharing scheme. A partial characterization of the cost-sharing schemes that correspond to group-strategyproof mechanisms in terms of a property called semi-cross-monotonicity is known; however, finding a complete characterization of cost-sharing schemes arising from group-strategyproof mechanisms remains an open question.

15.4 Cost Sharing via the Primal-Dual Schema

In Section 15.2.2, we discussed how a cost allocation in the approximate core of a game can be computed by solving an LP, and noted that for many combinatorial optimization games, this LP is equivalent to the dual of the standard LP relaxation of the problem and the cost shares correspond to the dual variables. In this section, we explain how a technique called the *primal-dual schema* can be used to compute cost shares that not only are in the approximate core of the game, but also satisfy the cross-monotonicity property, and hence can be used in the mechanism described in the previous section. The primal-dual schema is a standard technique in the field of approximation algorithms, where the focus is on computing an approximately optimal primal solution, and the dual variables (cost shares) are merely a by-product of the algorithm.

The idea of the primal-dual schema, which is often used to solve cost minimization problems, is to write the optimization problem as a mathematical program that can be relaxed into an LP (we refer to this LP as the primal LP). The dual of this LP gives a lower bound on the value of the optimal solution for the problem. Primal-dual algorithms simultaneously construct a solution to the primal problem and its dual. This is generally done by initially setting all dual variables to zero, and then gradually increasing these variables until some constraint in the dual program goes tight. This constraint hints at an object that can be *paid for* by the dual to be included in the primal solution. After this, the dual variables involved in the tight constraint are *frozen*, and the algorithm continues by increasing other dual variables. The algorithm ends when a complete solution for the primal problem is constructed. The analysis is based on proving that the values of the constructed primal and dual solutions are close to each other, and therefore they are both close to optimal.[3]

We will elaborate on two examples in this section: submodular games, where a simple primal-dual algorithm with no modification yields cross-monotonic cost-shares, and the facility location game, where extra care needs to be taken to obtain a cross-monotonic cost-sharing scheme. In the latter case, we introduce a rather general technique of using "ghost duals" to turn the standard primal-dual algorithm for the problem into an algorithm that returns a cross-monotonic cost-sharing scheme.

15.4.1 Submodular Games

Let us start with a definition of submodular games.

Definition 15.17 A cost-sharing game $(\mathcal{A}, c)$ is called a *submodular game* if the cost function c satisfies

$$\forall S, T \subseteq \mathcal{A}, \quad c(S) + c(T) \geq c(S \cup T) + c(S \cap T).$$

The above condition is equivalent to the condition of decreasing marginal cost, which says that for every two agents i and j and every set of agents $S \subset \mathcal{A} \setminus \{i, j\}$, the marginal cost of adding i to S (i.e., $c(S \cup \{i\}) - c(S)$) is no less than the marginal cost of adding i to $S \cup \{j\}$ (i.e., $c(S \cup \{i, j\}) - c(S \cup \{j\})$). Recall that we always assume $c(\emptyset) = 0$.

[3] In many primal-dual algorithms, a postprocessing step is required to bring the cost of the primal solution down. However, this step often does not change the cost shares.

Submodular games (also known as *concave* games) constitute an important class of cost-sharing games with many interesting properties. One example in this class is the multicast problem discussed in Section 14.2.2 of this book.

Consider a submodular game $(\mathcal{A}, c)$, and the LPs (15.2) and (15.1) as the primal and the dual programs for this game, respectively. It is not hard to see that by submodularity of c, the solution of the primal program is always $c(\mathcal{A})$, giving a trivial optimal solution for this LP. However, the dual LP (15.1) is nontrivial and its optimal solutions correspond to cost allocations in the core of the game. Let $\boldsymbol{\alpha}$ be a feasible solution of this LP. We say that a set $S \subseteq \mathcal{A}$ is *tight*, if the corresponding inequality in the LP is tight, i.e., if $\sum_{j \in S} \alpha_j = c(S)$. We need the following lemma to describe the algorithm.

Lemma 15.18 *Let $\boldsymbol{\alpha}$ be a feasible solution of the linear program (15.1). If two sets $S_1, S_2 \subseteq \mathcal{A}$ are tight, then so is $S_1 \cup S_2$.*

PROOF Since $\boldsymbol{\alpha}$ is feasible, we have $\sum_{j \in S_1 \cap S_2} \alpha_j \leq c(S_1 \cap S_2)$. This, together with the submodularity of c and tightness of S_1 and S_2, implies

$$\begin{aligned} c(S_1 \cup S_2) &\leq c(S_1) + c(S_2) - c(S_1 \cap S_2) \\ &\leq \sum_{j \in S_1} \alpha_j + \sum_{j \in S_2} \alpha_j - \sum_{j \in S_1 \cap S_2} \alpha_j \\ &= \sum_{j \in S_1 \cup S_2} \alpha_j \end{aligned}$$

Therefore, $S_1 \cup S_2$ is tight. □

Corollary 15.19 *There is a unique maximal tight set. It is simply the union of all the tight sets.*

We are now ready to state the algorithm that computes the cost shares. This algorithm is presented in Figure 15.3. Notice that by Lemma 15.18, when a new set goes tight, the new maximal tight set contains the old one, and therefore once an element $i \in T$ is included in the frozen set F, it will stay in this set until the end of the algorithm. Thus, the cost share α_j at the end of the algorithm is precisely the first time at which the element j is frozen. Furthermore, note that the algorithm never allows $\boldsymbol{\alpha}$ to become an infeasible solution of the LP (15.1), and stops only when the set T goes tight. Hence, the cost shares computed by the algorithm satisfy the budget balance and the core property. All that remains is to show that they also satisfy the cross-monotonicity property.

Theorem 15.20 *The cost sharing scheme defined by the algorithm* SubmodularCostShare *(Figure 15.3) is cross monotone.*

PROOF Let $T_1 \subset T_2 \subseteq \mathcal{A}$. We simultaneously run the algorithm for T_1 and T_2, and call these two runs the T_1-run and the T_2-run. It is enough to show that at any moment, the set of frozen elements in the T_1-run is a subset of that of the T_2-run. Consider a time t (i.e., the moment when all unfrozen cost shares in both runs are

```
Algorithm SubmodularCostShare

Input: submodular cost sharing game $(\mathcal{A}, c)$
        set $T \subseteq \mathcal{A}$ of agents that receive the service

Output: cost shares $\alpha_j$ for every $j \in T$

For every $j$, initialize $\alpha_j$ to 0.
Let $F = \emptyset$
While $T \setminus F \neq \emptyset$ do
    Increase all $\alpha_j$'s for $j \in T \setminus F$ at the same rate,
        until a new set goes tight.
    Let $F$ be the maximal tight set.
```

Figure 15.3. Algorithm for computing cost shares in a submodular game.

equal to t), and let $\boldsymbol{\alpha}^{\mathbf{l}}$ and F_l denote the values of the variables and the frozen set at this moment in the T_l-run, for $l = 1, 2$. We have

$$\begin{aligned} c(F_1 \cup F_2) &\leq c(F_1) + c(F_2) - c(F_1 \cap F_2) \\ &\leq \sum_{i \in F_1} \alpha_i^1 + \sum_{i \in F_2} \alpha_i^2 - \sum_{i \in F_1 \cap F_2} \alpha_i^1 \\ &= \sum_{i \in F_1 \setminus F_2} \alpha_i^1 + \sum_{i \in F_2} \alpha_i^2 \\ &\leq \sum_{i \in F_1 \cup F_2} \alpha_i^2, \end{aligned}$$

where the first inequality follows from submodularity of c, the second follows from the tightness of F_l with respect to $\boldsymbol{\alpha}^l$ ($l = 1, 2$) and the feasibility of $\boldsymbol{\alpha}^1$, and the last follows from the fact that for every $i \in F_1 \setminus F_2$, since $i \in T_1 \subset T_2$ and i is not frozen at time t in the T_2-run, we have $\alpha_i^2 = t \geq \alpha_i^1$. The above inequality implies that $F_1 \cup F_2$ is tight with respect to $\boldsymbol{\alpha}^2$. Since by definition F_2 is the maximal tight set with respect to $\boldsymbol{\alpha}^2$, we must have $F_1 \cup F_2 = F_2$. Hence, $F_1 \subseteq F_2$, as desired. □

15.4.2 The Facility Location Game

We now turn to our second example, the facility location game, and observe how the standard primal-dual scheme for this problem fails to satisfy cross-monotonicity.

Recall the LP formulation (15.3) of the facility location problem, and the observation that one can assume, without loss of generality, that in a solution to this program, we have $\beta_{ij} = \max(0, \alpha_j - d_{ij})$. In designing a primal-dual algorithm for the facility location game, we think of the quantity $\max(0, \alpha_j - d_{ij})$ as the *contribution* of agent j toward facility i, and say that a facility i is *tight* with respect to the dual solution $\boldsymbol{\alpha}$, if the total contribution i receives in $\boldsymbol{\alpha}$ equals its opening cost, i.e., if $\sum_{j \in \mathcal{A}} \max(0, \alpha_j - d_{ij}) = f_i$.

Following the general paradigm of the primal-dual schema, let us consider the following algorithm for computing cost shares in the facility location game: initialize the cost shares α_j to zero and gradually increase them until one of these two events occurs: a facility i goes tight, in which case, this facility is opened, and the cost shares of all agents j with positive contribution to i are frozen (i.e., no longer increased); or for an agent j and a facility i that is already opened, $\alpha_j = d_{ij}$, in which case the cost share of j is frozen. This process continues until all cost shares are frozen.

To illustrate this algorithm, consider the facility location game in Example 15.2 (Figure 15.1). In this example, at time 2, facility 2 goes tight as each of b and c makes one unit of contribution toward this facility. Therefore, the cost shares of b and c are frozen at 2. The cost share of the agent a continues to increase to 4, at which point facility 1 also goes tight and the algorithm stops. In this example, the cost allocation (4, 2, 2) computed by the algorithm is budget balanced and satisfies the core property. In fact, it is known that in every instance, with a postprocessing step that closes some of the open facilities, the cost of the primal solution can be brought down to at most 3 times the sum of cost shares, and therefore the cost shares are $\frac{1}{3}$-budget-balanced on every instance of the facility location game. However, unfortunately the cross-monotonicity property fails, as can be seen in the example in Figure 15.1. In this example, if only agents a and b are present, they will both increase their cost share to 3, at which point both facilities go tight and the algorithm stops. Hence, agent a has a smaller cost share in the set $\{a, b\}$ than in $\{a, b, c\}$.

Intuitively, the reason for the failure of cross-monotonicity in the above example is that without c, b helps a pay for the cost of facility 1. However, when c is present, she helps b pay for the cost of facility 2. This, in turn, hurts a, as b stops helping a as soon as facility 2 is opened. This suggests the following way to fix the problem: we modify the algorithm so that even after an agent is frozen, she continues to grow her *ghost* cost share. This ghost cost share is not counted toward the final cost share of the agent, but it can help other agents pay for the opening cost of facilities. For example, in the instance in Figure 15.1 when all three agents are present, even though agents b and c stop increasing their cost share at time 2, their ghost share continues to grow, until at time 3, facility 1 is opened with contributions from agents a and b. At this point, the cost share of agent a is also frozen and the algorithm terminates. The final cost shares will be 3, 2, and 2. The pseudo-code of this algorithm is presented in Figure 15.4. Variables $\boldsymbol{\alpha}'$ in this pseudo-code represent the ghost cost shares.

With this modification, it is an easy exercise to show that the cost shares computed by the algorithm are cross-monotone. However, it is not clear that the budget balance property is preserved. In fact, it is natural to expect that having ghost cost shares that contribute toward opening facilities in the primal solution but do not count toward the final cost shares could hurt the budget balance (see, i.e., Exercise 15.3). For the facility location problem, the following theorem shows that this is not the case.

Theorem 15.21 *The cost allocation computed by the algorithm* FLCostShare *(Figure 15.4) is $\frac{1}{3}$-budget balanced.*

```
Algorithm FLCostShare
Input: facility location game $(\mathcal{A}, c)$ defined by
           facility opening costs $f_i$ and distances $d_{ij}$
       set $T \subseteq \mathcal{A}$ of agents that receive the service
Output: cost shares $\alpha_j$ for every $j \in T$

For every $j$, initialize both $\alpha_j$ and $\alpha'_j$ to 0.
Let $F = \emptyset$.
While $T \setminus F \neq \emptyset$ do
    Increase all $\alpha_j$'s for $j \in T \setminus F$ and $\alpha'_j$'s for $j \in T$ at the same rate, until
        • for an unopened facility $i$, $\sum_{j \in T} \max(0, \alpha'_j - d_{ij}) = f_i$:
            in this case, open facility $i$, and
            add every agent $j$ with a positive contribution toward $i$ to $F$;
        • for an open facility $i$ and an agent $j$, $\alpha_j = d_{ij}$:
            in this case, add $j$ to $F$.
```

Figure 15.4. Algorithm for computing cost shares in the facility location game.

PROOF It is enough to show that for every instance of the facility location problem, it is possible to construct a solution whose cost is at most three times the sum of the cost shares computed by FLCostShare. To do this, we perform the following postprocessing step on the solution computed by FLCostShare. Let t_i denote the time at which facility i is opened by FLCostShare, and order all facilities that are opened by this algorithm in the order of increasing t_i's. We proceed in this order and for any facility i, check if there is any *open* facility i' that comes before i in this order and is within distance $2t_i$ of i. If such a facility exists, we close facility i; otherwise, we keep it open. After processing all facilities in this order, let $\mathcal{F}'$ denote the set of facilities that remain open and connect each agent in T to its closest facility in $\mathcal{F}'$. We now show that $\sum_{j \in T} \alpha_j$ is enough to pay for at least one third of the cost of this solution.

Let S_i denote the set of agents within distance t_i of facility i. First, observe that for any two facilities i and i' in $\mathcal{F}'$, S_i and $S_{i'}$ are disjoint. This is because if there is an agent j in $S_i \cap S_{i'}$, the distance between i and i' is at most $t_i + t_{i'} \leq 2\max(t_i, t_{i'})$, and therefore one of i and i' must have been closed in the above postprocessing step. To complete the proof, it is enough to show two statements. First, we show that for every facility $i \in \mathcal{F}'$, the cost shares of agents in S_i is enough to pay for at least a third of their distances to i (and hence, to their closest facility in $\mathcal{F}'$) plus the opening cost of i. Second, we show that each agent j that is not in $\cup_{i \in \mathcal{F}'} S_i$ can pay for at least one third of its distance to its closest facility in $\mathcal{F}'$.

To prove the first statement, note that for every facility $i \in \mathcal{F}'$, the ghost cost share of every agent contributing to i at the time of its opening is precisely t_i; hence, $\sum_{j \in S_i}(t_i - d_{ij}) = f_i$. Therefore, if we show that for every $j \in S_i$, $\alpha_j \geq t_i/3$, we would get $\sum_{j \in S_i} \alpha_j \geq \frac{1}{3}(f_i + \sum_{j \in S_i} d_{ij})$. Assume, for contradiction, that there is an agent j with $\alpha_j < t_i/3$ and consider the facility i_1 with $t_{i_1} = \alpha_j$ (i.e., the facility whose opening has caused the cost share of j to freeze). There must be a facility $i_2 \in \mathcal{F}'$ that is within distance $2t_{i_1}$ of i_1 (if $i_1 \in \mathcal{F}'$, we can let $i_2 = i_1$). Therefore, the distance between i and i_2 is at most $d_{ij} + d_{i_1 j} + 2t_{i_1} \leq t_i + 3\alpha_j < 2t_i$. This contradicts the assumption that $i \in \mathcal{F}'$, since i comes after i_2 in the ordering and $i_2 \in \mathcal{F}'$.

To show the second statement, consider an agent $j \in T \setminus \cup_{i \in \mathcal{F}'} S_i$, and let i be the facility with $t_i = \alpha_j$. There must be a facility i' in $\mathcal{F}'$ that is within distance $2t_i$ of i (i' can be the same as i). Therefore, the distance from j to its closest facility in $\mathcal{F}'$ is at most $d_{ij} + 2t_i \leq 3\alpha_j$. □

15.5 Limitations of Cross-Monotonic Cost-Sharing Schemes

As we saw in Section 15.3, a cost-sharing scheme that is cross-monotone also satisfies the core property. As a result, for any combinatorial cost-sharing game, an upper bound on the budget balance factor of cross-monotonic cost-sharing schemes can be obtained using Theorem 15.8 and the integrality gap examples of the corresponding LP. As we will see in this section, for many combinatorial optimization games, the cross-monotonicity property is strictly stronger than the core property, and better upper bounds on the budget balance factor of such games can be obtained using a technique based on the probabilistic method.

The high-level idea of this technique is as follows: Fix any cross-monotonic cost-sharing scheme. We explicitly construct an instance of the game and look at the cost-sharing scheme on various subsets of this instance. We need to argue that there is a subset S of agents such that the total cost shares of the elements of S is small compared to the cost of S. This is done using the probabilistic method: we pick a subset S at random from a certain distribution and show that in expectation, the ratio of the recovered cost to the cost of S is low. Therefore, there is a manifestation of S for which this ratio is low. To bound the expected value of the sum of cost shares of the elements of S, we use cross-monotonicity and bound the cost share of each agent $i \in S$ by the cost share of i in a substructure T_i of S. Bounding the expected cost share of i in T_i is done by showing that for every substructure T, every $i \in T$ has the same probability of occurring in a structure S in which $T_i = T$. This implies that the expected cost share of i in T_i (where the expectation is over the choice of S) is at most the cost of T_i divided by the number of agents in T_i. Summing up these values for all i gives us the desired bound.

In the following, we show how this technique can be applied to the facility location problem to show that the factor $1/3$ obtained in the previous section is the best possible. We start by giving an example on which the algorithm FLCostShare in Figure 15.4 recovers only a third of the cost. This example will be used as the randomly chosen structure in our proof.

Lemma 15.22 *Let $\mathcal{I}$ be an instance of the facility location problem consisting of $m + k$ agents $a_1, \ldots, a_m, a'_1, \ldots, a'_k$ and m facilities $f_1, \ldots, f_m$ each of opening cost 3. For every i and j, the connection costs between f_i and a_i and between*

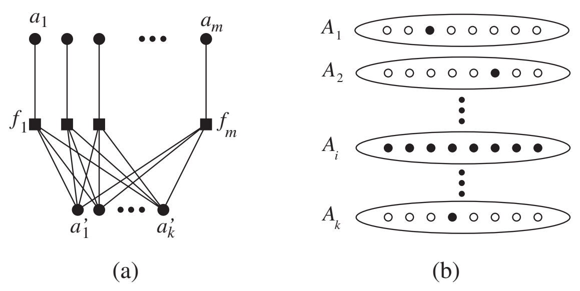

Figure 15.5. Facility location sample distribution.

f_i and a'_j are all 1, and other connection costs are obtained by the triangle inequality. See Figure 15.5a. Then if $m = \omega(k)$ and k tends to infinity, the optimal solution for $\mathcal{I}$ has cost $3m + o(m)$.

PROOF The solution which opens just one facility, say f_1, has cost $3m + k + 1 = 3m + o(m)$. We show that this solution is optimal. Consider any feasible solution that opens f facilities. The first opened facility can cover $k + 1$ agents with connection cost 1. Each additional facility can cover 1 additional client with connection cost 1. Thus, the number of agents with connection cost 1 is $k + f$. The remaining $m - f$ agents have connection cost 3. Therefore, the cost of the solution is $3f + k + f + 3(m - f) = 3m + k + f$. As $f \geq 1$, this shows that any feasible solution costs at least as much as the solution we constructed. □

Theorem 15.23 *Any cross-monotonic cost-sharing scheme for facility location is at most* $1/3$*-budget-balanced.*

PROOF Consider the following instance of the facility location problem. There are k sets $A_1, \ldots, A_k$ of m agents each, where $m = \omega(k)$ and $k = \omega(1)$. For every subset B of agents containing exactly one agent from each A_i ($|B \cap A_i| = 1$ for all i), there is a facility f_B with connection cost 1 to each agent in B. The remaining connection costs are defined by extending the metric, that is, the cost of connecting agent i to facility f_B for $i \notin B$ is 3. The facility opening costs are all 3.

We pick a random set S of agents in the above instance as follows: Pick a random i from $\{1, \ldots, k\}$, and for every $j \neq i$, pick an agent a_j uniformly at random from A_j. Let $T = \{a_j : j \neq i\}$ and $S = A_i \cup T$. See Figure 15.5b for an example. It is easy to see that the set S induces an instance of the facility location problem almost identical to the instance $\mathcal{I}$ in Lemma 15.22 (the only difference is that here we have more facilities, but it is easy to see that the only relevant facilities are the ones that are present in $\mathcal{I}$). Therefore, the cost of the optimal solution on S is $3m + o(m)$.

We show that for any cross-monotonic cost-sharing scheme ξ, the average recovered cost over the choice of S is at most $m + o(m)$ and thus conclude that there is some S whose recovered cost is at most $m + o(m)$. We start by bounding the expected total cost share using linearity of expectation and cross-monotonicity:

$$\begin{aligned} \mathrm{E}_S\left[\sum_{a\in S}\xi(a,S)\right] &= \mathrm{E}\left[\sum_{a\in A_i}\xi(a,S)\right] + \mathrm{E}\left[\sum_{j\neq i}\xi(a_j,S)\right] \\ &\leq \mathrm{E}\left[\sum_{a\in A_i}\xi(a,\{a\}\cup T)\right] + \mathrm{E}\left[\sum_{j\neq i}\xi(a_j,T)\right]. \end{aligned}$$

Notice that the set T has a facility location solution of cost $3 + k - 1$ and thus by the budget-balance condition the second term in the above expression is at most $k + 2$. The first term in the above expression can be written as $m\mathrm{E}_{S,a}[\xi(a, \{a\} \cup T)]$, where the expectation is over the random choice of S and the random choice of a from A_i. This is equivalent to the following random experiment: From each A_j, pick an agent a_j uniformly at random. Then pick i from $\{1, \dots, k\}$ uniformly at random and let $a = a_i$ and $T = \{a_j : j \neq i\}$. From this description it is clear that the expected value of $\xi(a, \{a\} \cup T)$ is equal to $\frac{1}{k}\sum_{j=1}^{k}\xi(a_j, \{a_1, \dots, a_k\})$. This, by the budget-balance property and the fact that $\{a_1, \dots, a_k\}$ has a solution of cost $k + 3$, cannot be more than $\frac{k+3}{k}$. Therefore,

$$\mathrm{E}_S\left[\sum_{a\in S}\xi(a,S)\right] \leq m\left(\frac{k+3}{k}\right) + (k+2) = m + o(m), \qquad (15.6)$$

when $m = \omega(k)$ and $k = \omega(1)$. Therefore, the expected value of the ratio of recovered cost to total cost tends to $1/3$. □

15.6 The Shapley Value and the Nash Bargaining Solution

One of the problems with the notion of core in cost-sharing games is that it rarely assigns a unique cost allocation to a game: as illustrated in Example 15.4, the core of a game is often either empty (making it useless in deciding how the cost of a service should be shared among the agents), or contains more than one point (making it necessary to have a second criterion for choosing a cost allocation). In this section, we study a solution concept called the Shapley value that assigns a single cost allocation to any given cost-sharing game. We also discuss a solution concept known as the Nash bargaining solution for a somewhat different but related framework for surplus sharing. In both cases, the solution concept can be uniquely characterized in terms of a few natural axioms it satisfies. These theorems are classical examples of the *axiomatic approach* in economic theory.

Both the Shapley value and the Nash bargaining solution are widely applicable concepts. For example, an application of the Shapley value in combination with the Moulin mechanism to multicasting is discussed elsewhere in this book (see Section 14.2.2). Also, the Nash solution is related to Kelly's notion of proportional fairness discussed in Section 5.12, and the Eisenberg-Gale convex program of Section 6.2.

15.6.1 The Shapley Value

Consider a cost-sharing game defined by the set $\mathcal{A}$ of n agents and the cost function c. A simple way of allocating the cost $c(\mathcal{A})$ among all agents is to order the agents in some order, say $a_1, a_2, \ldots, a_n$, then proceed in this order and charge each agent the marginal cost of adding her to the serviced set. In other words, the first agent a_1 will be charged her stand-alone cost $c(\{a_1\})$, the second agent a_2 will be charged $c(\{a_1, a_2\}) - c(\{a_1\})$, and so on. This method is called an *incremental cost sharing*.

A problem with the method described above is that it is not anonymous, i.e., the ordering of the agents makes a difference in the amount they will be charged. The Shapley value fixes this problem by taking a *random* ordering of the agents picked uniformly from the set of all $n!$ possible orderings, and charging each agent her *expected* marginal cost in this ordering. Since for any agent $i \in \mathcal{A}$ and any set $S \subseteq \mathcal{A} \setminus \{i\}$ with $|S| = s$, the probability that the set of agents that come before i in a random ordering is precisely S is $s!(n-1-s)!/n!$, the Shapley value can be defined by the following formula:

$$\text{For each agent } i, \quad \phi_i(c) = \sum_{s=0}^{n-1} \frac{s!(n-1-s)!}{n!} \sum_{S \subseteq \mathcal{A}\setminus\{i\}, |S|=s} (c(S \cup \{i\}) - c(S)),$$

where $\phi_i(c)$ indicates the cost share of $i \in \mathcal{A}$ in the cost-sharing game $(\mathcal{A}, c)$. As the following example shows, the cost sharing given by the Shapley value need not be in the core of the game, even if the core is nonempty.

Example 15.24 Consider the facility location game defined in Example 15.2. The Shapley values in this game are as follows:

$$\phi_a = \frac{1}{3} \times 4 + \frac{1}{6} \times 3 + \frac{1}{6} \times 4 + \frac{1}{3} \times 4 = \frac{23}{6}$$
$$\phi_b = \frac{1}{3} \times 3 + \frac{1}{6} \times 2 + \frac{1}{6} \times 1 + \frac{1}{3} \times 1 = \frac{11}{6}$$
$$\phi_c = \frac{1}{3} \times 3 + \frac{1}{6} \times 3 + \frac{1}{6} \times 1 + \frac{1}{3} \times 2 = \frac{7}{3}$$

This cost allocation is not in the core of the game, since $\phi_b + \phi_c = \frac{25}{6} > 4 = c(\{b, c\})$. This is despite the fact that, as we saw in Example 15.4, the core of this game is nonempty.

However, for submodular games, it is known that any incremental cost-sharing, and therefore the Shapley value (which is a linear combination of incremental cost-sharing methods), is in the core of the game. In fact, it can be shown that in this class of games, the Shapley value is cross-monotone (see Exercise 15.2), making it useful in the design of group-strategyproof mechanisms using Moulin's mechanism of Section 15.3.[4] This

[4] It is worth noting that since the Shapley value is defined in terms of a formula comprising of exponentially many points of the function $c(\cdot)$, evaluating it is computationally hard in general. However, when the cost function c is submodular, random sampling can be used to approximate the Shapley values to within an arbitrary degree of accuracy.

is used in Section 14.2.2 of this book, in an application to the multicast problem. Many other applications of the Shapley value, as well as various generalizations (to settings such as NTU games or games with nonbinary demands), are extensively studied in the economic literature.

15.6.2 An Axiomatic Characterization of the Shapley Value

In his original paper, Shapley introduced what is now known as the Shapley value as the unique value satisfying the three properties defined below.

Definition 15.25 Fix a set $\mathcal{A}$ of n agents. A *value* is a function that assigns to each cost function c a vector $\phi(c) \in \mathbb{R}^n$ of nonnegative numbers. Three properties of values are defined as follows.

- *Anonymity*: Changing the names of the agents does not change their cost shares. Formally, ϕ satisfies anonymity if for every permutation π of $\mathcal{A}$ and every cost function c, $\phi_{\pi_i}(\pi(c)) = \phi_i(c)$ for every $i \in \mathcal{A}$.
- *Dummy*: An agent who does not add to the cost should not be charged anything. More precisely, if for every set $S \subset \mathcal{A} \setminus \{i\}$, $c(S) = c(S \cup \{i\})$, then $\phi_i(c) = 0$.
- *Additivity*: For every two cost functions c_1 and c_2, $\phi(c_1 + c_2) = \phi(c_1) + \phi(c_2)$, where $c_1 + c_2$ is the cost function defined by $(c_1 + c_2)(S) = c_1(S) + c_2(S)$.

Theorem 15.26 *The Shapley value is the unique value satisfying anonymity, dummy, and additivity.*

The above theorem, whose proof is omitted here, is an example of the *axiomatic* method in the economic theory, whose goal is to find (or prove the nonexistence of) solution concepts that satisfy certain sets of desirable axioms, or characterize known solution concepts in terms of axioms they satisfy. Two other prominent examples of axiomatic results are Nash's theorem on bargaining (presented in the next section; this result is considered a starting point for the axiomatic approach in economic theory), and Arrow's impossibility result in the social choice literature. One example where this framework is applied in computer science is the axiomatic characterization of the *PageRank* algorithm for ranking Web search results.

15.6.3 The Nash Bargaining Solution

The bargaining problem studies a situation where two or more agents need to select one of the many possible outcomes of a joint collaboration. Examples include wage negotiation between an employer and a potential employee, or trade negotiation between two countries. Each party in the negotiation has the option of leaving the table, in which case the bargaining will result in a *disagreement outcome*. More formally, a bargaining game for two players (the case of more players is similar) is given by a set $X \in \mathbb{R}^2$, along with a *disagreement point* $d \in X$. Each point in X corresponds to one outcome of the bargaining, and specifies the utility of each player for this outcome. The point d specifies the utility of each player for the disagreement outcome. As adding or subtracting a value to the utility of an individual does not change her relative preferences, we assume, without loss of generality, that $d = (0, 0)$. Furthermore, we assume that the set X is convex and compact. Note that convexity of X is without loss of generality, if an outcome is allowed to be a probability distribution over *pure* outcomes. Furthermore, we assume X contains at least one point whose coordinates are both positive (i.e., both parties have some incentive to negotiate).

The above model for bargaining was first defined and studied by Nash. Note that an NTU cooperative game can be considered an extension of the bargaining model, where in addition to the outcome of individual deviations (the disagreement point), the outcome of group deviations are also given. Nash's bargaining theorem gives a characterization of a *solution* for the bargaining game in terms of axioms it satisfies. Formally,

Definition 15.27 A solution for the bargaining game (also known as a *social choice function*) is a function that assigns to each set X satisfying the above properties a single point $\phi(X) \in X$. We define four properties of a solution as follows:

- *Pareto Optimality*: $\phi(X)$ is a Pareto optimal point in X, i.e., there is no point $p \in X$ with $p > \phi(X)$, coordinate-wise.
- *Symmetry*: If the set X is symmetric, then $\phi(X) = (u, u)$ for some $u \in \mathbb{R}$.
- *Scale Independence*: The solution is independent of the scale used to measure individual utilities; i.e., if X' is obtained from X by multiplying all utilities of the i'th player by λ_i, then $\phi(X')$ can be obtained from $\phi(X)$ by multiplying the i'th coordinate by λ_i.
- *Independence of Irrelevant Alternatives*: If $Y \subset X$ and $\phi(X) \in Y$, then $\phi(Y) = \phi(X)$.

We now state Nash's bargaining theorem. The proof of this theorem is simple, and is omitted here.

Theorem 15.28 *There is a unique solution for bargaining games satisfying Pareto optimality, symmetry, scale independence, and independence of irrelevant alternatives. This solution assigns to each set X a point* (u_1, u_2) *maximizing* $u_1 u_2$.

Nash's theorem gives one example of what is called a *collective utility function*. A collective utility function is a function that aggregates the utilities of individuals into a single number indicating the utility of the *society*. Classical examples of collective utility functions are the utilitarian function (which simply adds up the individual utilities), the egalitarian function (which takes the minimum of individual utilities), and the Nash function (which takes the product of the utilities).

15.7 Conclusion

In this chapter, we reviewed some of the basic notions (such as the core, cross-monotonicity, group-strategyproof mechanisms, Shapley value, and the Nash bargaining solution) and classical results on cost and surplus sharing. We observed that the algorithmic questions regarding computing cost shares are closely tied to the LP formulation of the corresponding optimization problem, and explained how standard LP-based techniques developed in the field of approximation algorithms can be used to tackle such questions.

There is also a potential for contributions in the other direction. For many combinatorial optimization problems, thinking in terms of the cost-sharing problem (i.e., the dual problem) instead of the primal can shed new light on the problem. In the facility location example discussed in Section 15.4.2, the proof of Theorem 15.21 gives an approximation algorithm different from the standard primal-dual algorithm for the problem. As it turns out, in this case the algorithm was known before, but Theorem 15.21 gives a new primal-dual interpretation of this algorithm. For the Steiner forest problem, the search for a cross-monotonic cost-sharing scheme has resulted in a new 2-approximation algorithm, and a stronger LP relaxation for the problem. In fact, for most combinatorial optimization problems, LP (15.2) is at least as strong an LP formulation as the standard LP relaxation; i.e., it gives at least as good a lower bound on the value of the optimal solution. These LPs are equivalent for some problems, as we saw in Section 15.2.2 for the facility location problem. However, for many other problems, such as the well-studied Steiner tree problem, this appears not to be the case. Therefore, one possible approach to obtain stronger LP relaxations (which could lead to better approximation algorithms) for such problems is to start from (15.2) and try to relax this program into one that can be solved in polynomial time. In the case of the Steiner tree problem, the integrality gap of LP (15.2) seems to be related to the long open question on the integrality gap of the bidirected LP relaxation of this problem.

Another way the economic approach to cost sharing can contribute to the theory of algorithms is by providing new perspectives and new problems. For example, the axiomatic approach explained in Section 15.6 seems to be a suitable tool for studying properties of heuristic algorithms. One notable example is the axiomatic characterization of the popular web ranking algorithm PageRank. Also, the field of combinatorial optimization almost exclusively deals with problems whose objective is to minimize the total cost, or maximize the total benefit, which, according to the terminology introduced in Section 15.6.3, corresponds to the utilitarian collective utility function. However, the field of social choice suggests other objective functions, which can lead to new challenging algorithmic questions. One notable example is the Santa Claus problem, which seeks to optimize the egalitarian objective in a simple scheduling model. Also, many of the algorithmic results presented in Chapter 5 for Fisher markets (where the Eisenberg–Gale convex program shows that the market equilibrium corresponds to the point maximizing the Nash collective utility function) can be viewed in this light.

15.8 Notes

Sections 15.1 and 15.2. The notion of a cooperative game was first proposed by von Neumann and Morgenstern (1944). The notion of the core was first introduced by

Gillies (1959). Theorem 15.6 was independently discovered by Bondareva (1963) and Shapley (1967). Theorem 15.11 is due to Scarf (1967). Deng et al. (1997) observed the connection between the core of many combinatorial optimization games and the integrality gap of the corresponding LP. Goemans and Skutella (2000) showed this connection for the facility location game, and proved that deciding whether the core of a facility location game is nonempty is NP-complete. The best lower and upper bound on the integrality gap of LP (15.3) are $\frac{1}{1.52}$, due to Mahdian et al. (2006), and $\frac{1}{1.463}$, due to Guha and Khuller (1999) See Immorlica et al. (2006) for an example of a problem modeled using NTU games.

Section 15.3. For a discussion of the NPT, VP, and CS properties of cost sharing mechanisms see Moulin (1999) and Moulin and Shenker (2001). In our definition of group-strategyproof mechanisms, we did not allow side payments between members of a coalition. For a discussion of mechanism design in a setting where collusion with side payments is allowed, see Goldberg and Hartline (2005). This cross-monotonicity property for cost sharing is similar to the population monotonicity property introduced by Thomson (1983, 1995) in the context of bargaining. For cooperative games, this notion was first introduced by Sprumont (1990). The mechanism $\mathcal{M}_\xi$ and Theorem 15.16 are due to Moulin (1999), where he also proves a converse to this theorem for submodular games. Examples on the connection between group-strategyproof mechanisms and cost-sharing schemes, and a partial characterization of such mechanisms are due to Immorlica et al. (2005).

Sections 15.4 and 15.5. For a general introduction to the primal-dual schema from the perspective of approximation algorithms, see the excellent book by Vazirani (2001). The cost-sharing scheme presented in Section 15.4 for submodular games is due to Dutta and Ray (1989). This scheme was formulated as a primal-dual algorithm and generalized to an algorithm that can increase the dual variables at different rates by Jain and Vazirani (2002). Both Dutta and Ray (1989) Jain and Vazirani (2002) also prove several fairness properties of their cost-sharing schemes. The technique of using ghost duals and its application to the facility location problem (algorithm in Figure 15.4) and single-source rent-or-buy problem are due to Pál and Tardos (2003). The proof of their result on the facility location problem (Theorem 15.21) is based on an algorithm that is originally due to Mettu and Plaxton (2000). The first (non-cross-monotonic) primal-dual algorithm for the facility location problem is due to Jain and Vazirani (2001). The probabilistic technique presented in Section 15.5 and its application to several problems including facility location, vertex cover, and set cover are due to Immorlica et al. (2005). Könemann et al. (2007) gave a 1/2-budget-balanced mechanism, together with a matching upper bound for the Steiner forest problem.

Section 15.6. The Shapley value and its axiomatic characterization (Theorem 15.26) are due to Shapley (1953). In the same paper, Shapley shows that for convex games (which correspond to submodular games in the context of cost sharing) the Shapley value is in the core. The application of Shapley values to the multicast problem is due to Feigenbaum et al. (2000) and is explained in detail in Chapter 14. For other applications of the Shapley value, see the book edited by Roth (1988) or the survey by Winter (2002). The generalization of the Shapley value to games with nonbinary demand is due to Aumann and Shapley (1974). See the survey by McLean (1994) for various generalizations to NTU games. The result on the computation of Shapley values for submodular games is due to Mossel and Saberi (2006). The axiomatic result of Arrow is given in Arrow (1959). Axiomatic characterizations of PageRank (Page et al., 1999) are given by Palacois-Huerta and Volij (2004) and by Altman and Tennenholtz (2005). We refer the reader to the excellent survey by Moulin (2002) for further information on the axiomatic approach to cost sharing. Theorem 15.28 is proved in a seminal paper by Nash (1950). See Moulin (1988) for further discussion of this theorem and its generalization to more than two players. See Moulin (1988, 2003) for a discussion of various collective utility functions and social choice rules. For more information on the Santa Claus problem see Bansal and Sviridenko (2006) and Asadpour and Saberi (2006).

Acknowledgment

We would like to thank Anna Karlin, Hervé Moulin, and Nicole Immorlica for their valuable comments.

Bibliography

A. Altman and M. Tennenholtz. Ranking systems: The PageRank axioms. In *Proc. 6th ACM Conf. Electronic Commerce*, pp. 1–8, 2005.

K.J. Arrow. Rational choice functions and orderings. *Econometrica*, 26:121–127, 1959.

A. Asadpour and A. Saberi. An approximation algorithm for max-min fair allocation of indivisible goods. In *Proc. 39th Annual ACM Symp. Theory of Computing*, 2007.

R.J. Aumann and L.S. Shapley. *Values of Non-Atomic Games*. Princeton University Press, 1974.

N. Bansal and M. Sviridenko. The Santa Claus problem. In *Proc. 38th Annual ACM Symp. Theory of Computing*, 2006.

O.N. Bondareva. Some applications of linear programming to cooperative games. *Problemy Kibernetiki*, 1963.

X. Deng, T. Ibaraki, and H. Nagamochi. Algorithms and complexity in combinatorial optimization games. In *Proc. 8th ACM Symp. on Discrete Algorithms*, 1997.

B. Dutta and D. Ray. A concept of egalitarianism under participation constraints. *Econometrica*, 57(3):615–635, May 1989.

J. Feigenbaum, C. Papadimitriou, and S. Shenker. Sharing the cost of multicast transmission. *Proc. 32nd Annual ACM Symp. Theory of Computing*, 2000.

D.B. Gillies. Solutions to general non-zero-sum games. In A. W. Tucker and R. D. Luce, editors, *Contributions to the Theory of Games*, volume IV, pp. 47–85. Princeton University Press, 1959.

M.X. Goemans and M. Skutella. Cooperative facility location games. *Symp. on Discrete Algorithms*, 2000.

A.V. Goldberg and J.D. Hartline. Collusion-resistant mechanisms for single-parameter agents. In *Proc. 16th ACM Symp. Discrete Algorithms*, pp. 620–629, 2005.

S. Guha and S. Khuller. Greedy strikes back: Improved facility location algorithms. *J. Algorithms*, 31:228–248, 1999.

N. Immorlica, K. Jain, and M. Mahdian. Game-theoretic aspects of designing hyperlink structures. In *Proc. 2nd Workshop on Internet and Network Economics (WINE)*, LNCS 4286:150–161, Springer, 2006.

N. Immorlica, M. Mahdian, and V.S. Mirrokni. Limitations of cross-monotonic cost-sharing schemes. In *Proc. 16th ACM Symp. Discrete Algorithms*, 2005.

K. Jain and V.V. Vazirani. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation. *J. ACM*, 48:274–296, 2001.

K. Jain and V.V. Vazirani. Equitable cost allocations via primal-dual-type algorithms. In *Proc. 34th Annual ACM Symp. Theory of Computing*, pp. 313–321, 2002.

J. Könemann, S. Leonardi, G. Schäfer, and S. van Zwam. From primal-dual to cost shares and back: A group-strategyproof mechanism for the Steiner forest game. to appear in *SIAM J. Computing*, 2007.

M. Mahdian, Y. Ye, and J. Zhang. Approximation algorithms for metric facility location problems. *SIAM J. Comp.*, 36(2):411–432, 2006.

R.P. McLean. Values of non-transferable utility games. In R.J. Aumann and S. Hart, editors, *Handbook of Game Theory with Economic Applications*, 2:2077–2120. Elseveir Science Publishers B.V., 1994.

R.R. Mettu and G. Plaxton. The online median problem. In *Proc. 41st Symp. on Fdns. of Computer Science*, pp. 339–348, 2000.

E. Mossel and A. Saberi. On efficiently computing the Shapley value of a game. unpublished manuscript, 2006.

H. Moulin. *Axioms of Cooperative Decision Making*. Cambridge University Press, 1988.

H. Moulin. Incremental cost sharing: Characterization by coalition strategy-proofness. *Soc. Choice Welfare*, 16:279–320, 1999.

H. Moulin. Axiomatic cost and surplus sharing. In K. J. Arrow, A. K. Sen, and K. Suzumura, editors, *Handbook of Social Choice and Welfare*, 1:289–357. Elseveir Science Publishers B.V., 2002.

H. Moulin. *Fair Division and Collective Welfare*. MIT Press, 2003.

H. Moulin and S. Shenker. Strategyproof sharing of submodular costs: Budget balance vs. efficiency. *Econ. Theory*, 18:511–533, 2001.

J.F. Nash. The bargaining problem. *Econometrica*, 28:155–62, 1950.

L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web. Technical Report SIDL-WP-1999-0120, Stanford CS Department, 1999.

M. Pál and É. Tardos. Group strategyproof mechanisms via primal-dual algorithms. In *Proc. 44th Annual IEEE Symp. on Fdns. of Computer Science*, pp. 584–593, 2003.

I. Palacois-Huerta and O. Volij. The measurement of intellectual influence. *Econometrica*, 72(3):963–977, May 2004.

A.E. Roth, editor. *The Shapley Value*. Cambridge University Press, 1988.

H.E. Scarf. The core of an n-person game. *Econometrica*, 35(1):50–69, 1967.

L.S. Shapley. A value for n-person games. In H. Kuhn and A.W. Tucker, editors, *Contributions to the Theory of Games*, 2:307–317. Princeton University Press, 1953.

L.S. Shapley. On balanced sets and cores. *Naval Res. Logistics Q.*, 14:453–460, 1967.

Y. Sprumont. Population monotonic allocation schemes for cooperative games with transferable utility. *Games Econ. Behav.*, 2:378–394, 1990.

W.L. Thomson. Problems of fair division and the egalitarian solution. *J. Econ. Theory*, 31:211–226, 1983.

W.L. Thomson. Population-monotonic allocation rules. In W.A. Barnett, H. Moulin, M. Salles, and N.J. Schofield, editors, *Social Choice, Welfare and Ethics*, 2:79–124. Cambridge University Press, 1995.

V.V. Vazirani. *Approximation Algorithms*. Springer-Verlag, Berlin, 2001.

J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. John Wiley and Sons, 1944.

E. Winter. The Shapley value. In R.J. Aumann and S. Hart, editors, *Handbook of Game Theory*. North-Holland, 2002.

Exercises

15.1 Consider a setting with n agents and m goods where each agent is endowed with a bundle of goods and a linear utility function that specifies the utility that this agent derives from consuming a bundle (this is the same as the linear Arrow–Debreu markets defined in Section 5.10 of this book). The value of a coalition S of agents in this model can be defined as the maximum total utility that agents in S can derive by optimally redistributing their endowments. Model this setting as a TU game. Does this game always have a nonempty core?

15.2 Prove that for submodular cost-sharing games, the Shapley value is cross-monotone.

15.3 In the vertex cover game, agents correspond to edges in a graph, and the cost of a set S of agents is the minimum size of a set of vertices that contains at least one of the endpoints of each edge in S. A simple primal-dual approach gives a 2-approximation algorithm for this problem. Modify this algorithm using the idea of ghost cost shares to obtain a cross-monotonic cost-sharing scheme. Find examples where this scheme fails to extract a constant fraction of the cost of the solution. Use this example, together with the technique explained in Section 15.5, to prove that no cross-monotonic cost-sharing scheme for this game is $\Omega(1)$-budget-balanced.